**Nginx Server Setup & Configurations**

**Environment Setup**

Installation VirutalBox

Installation of Vagrant

Install Ubuntu 18.4 using above technology

**Installing Nginx Server (06)**

1. change to root user using command ( **sudo su -** )

2. update and upgrade the ubuntu using command ( **apt-get update & apt-get upgrade**)

3. install nginx using command (**apt-get install nginx**)

4. confirm nginx installation by using command (**nginx -v**)

5. check nginx status using command ( **service nginx status or systemctl status nginx** )

6. check nginx status using browser (**192.168.0.3**) from host system

7. to make sure nginx is running (**s**

**Familiar with Nginx Files & Directories (07)**

/etc/nginx holds the configuration for entire nginx, Installation directory

/etc/nginx nginx.conf main configuration file

conf.d here we will store server configuration file

sites-available here we will store server configuration file

sites-enabled here we will store server configuration file

sites-available/default example configuration are stored here

vhost and server configuration are same terms

Below directories are outside of nginx installation directory

/var/log/nginx error log goes here

/var/www/ store actual file to severed.

**Useful Nginx commands (08)**

systemctl status nginx show nginx status or run with --no-pager switch

systemctl start nginx start nginx service

systemctl stop nginx stop nginx service

systemctl reload nginx reload the nginx service

nginx -h give help related to nginx

nginx -t check the configuration file and show status

nginx -T check the nginx configuration

after changing in configuration file, we need to check the configuration file using nginx -t and then use systemctl reload nginx to reload the configuration file.

**Working with main nginx configuration file ( /etc/nginx/nginx.config) (09)(10)(11)**

we should avoid any changes in this file, only read for information purpose

there are two place where we will put additional nginx configuration files

in **/etc/nginx/config.d** here file extension is not required

or

in **/etc/nginx/site-enabled** here file extension is required and link to sites-available

**Creating First Server Configuration (010)**

**Method No. 1 (/etc/nginx/conf.d/)**

1. remove the default configuration located in **/etc/nginx/sites-enabled/default.conf** using unlink

command to remove the linking

2. now create a file with **wisdompetmed.local.conf** in /**etc/nginx/config.d** locations and add the following configuration to file

server {

listen 80 default\_server;

server\_name wisdompetmed.local [www.wisdompetmed.local](http://www.websitename.local/);

index index.html index.htm index.php;

root /var/www/wisdompetmed.local;

}

5. now test the configuration of nginx using command **nginx -t** commands

6. reload the configuration by using **systemctl nginx reload**

7. create a directory with name wisdompetmed.local in /var/www/ location

8. create index.html file in /var/www/wisdompetmed .local with name index.html with following

contents

<h1> site coming soon </h1>

9. now test the server using browser

10. copy the files, directories, images to /var/www/wisdompetmed.local directory

11. set the permission, so that our site is secure with following command

find . -type -f -exec chmod 644 {} \;

find . -type -d -exec chmod 755 {} \;

**Method No. 1**

1. remove the default configuration located in /etc/nginx/sites-enabled/default.conf using unlink

command to remove the linking

2. now create a file with sitename.local.conf in /etc/nginx/sites-available locations and add the following configuration to file

server {

listen 80 default\_server;

server\_name sitename.local [www.sitename.local](http://www.sitename.local/) [www.sitename.com](http://www.sitename.com/);

index index.html;

root /var/www/sitename.local;

}

5. create a system link using command ln -s /etc/nginx/sites-available in sites-enabled directory

6. now test the configuration of nginx using command nginx -t commands

6. reload the configuration by using systemctl nginx reload

7. create a directory with name wisdompetmed.local in /var/www/ location

8. create a index.html file in /var/www/wisdompetmed.local with name index.html with following

contents

<h1> site coming soon </h1>

**Adding files to website (012)**

unzip the exercise file and copy to /var/www/wisdompetmed.local directory

**Setting Files & Directories Permissions (012)**

find path\_directory -type f -exec chmod 644 {} \;

find path\_directory -type d -exec chmod 755 {} \;

**Adding Location Directive to configuration File (013) & Configuring Logs (014)**

Location directive allow us to extend the configuration file based on URI, that make the request processed by server.

Configure Locations or Adding location directives in nginx configuration

location configuration directive allow us to extend our configurations base on URI

location directive come under server directives. It can be nested as well

first nginx will match exact location , then prefix and at last regular expression

server {

listen 80 default\_server;

server\_name website\_name.local [www.websitename.local](http://www.websitename.local/);

index index.html index.htm index.php;

root /var/www/wisdompetmed.local;

access\_log /var/log/nginx/widsompetmed.access.log;

error\_log /var/log/nginx/widsompetmed.error.log;

location /

{

try\_files $uri $uri/ =404; # root will check for uri , uri/ if no match then 404 page will display

}

location /images

{

autoindex on; # allow browser to list the contents of that directory

access\_loglog /var/log/nginx/widsompetmed.img.access.log;

error\_log /var/log/nginx/widsompetmed.img.error.log;

}

error\_page 404 /404.html;

location = /404.html

{

internal;

}

error\_page 500 502 503 504 /50x.html;

location = /50x.html

{

internal; # this is internal redirect

}

# this location is used for fake server side error, not required in production

location = /500

{

fastcgi\_pass unix: /this/will/fail;

}

}

**Adding Error log configuration to configuration File (014)**

add the below directive to server directive for recording errors in site.

access\_loglog /var/log/nginx/wisdompetmed.access.log;

error\_log /var/log/nginx/wisdompetmed.error.log;

**for i in {1..10}; do curl localhost > /dev/null; done**

above line will generate 10 request to our site.

we can add above directive to any block to record the error related to that specific location block

**Troubleshooting nginx (015)**

1. make sure nginx service is running

2. check configuration error using command nginx -t

3. check the root address and server directive spellings, check typo

error in configuration file

4. reload the nginx configuration using systemctl reload nginx commands

5. verify that port are open using commands

sudo lsof -P -n -n -i :80 -i :443 |grep LISTEN

OR

sudo netstat -plan | grep nginx

6. check the logs if still problem , check log using commands

**tail -f /var/log/nginx/wisdompetmed.error.log**

**LAMP Stack (016)**

**LAMP stack consist on Linux, Apache, MySQL and PHP**

**Installing PHP (017)**

to install php run the follow commands

sudo apt-get update

sudo apt-get install php-fpm php-mysql

check php version to make sure php is install successfully

php --version

make sure the fast cgi service is up & running

systemctl status php7.2-fpm

**now Configure nginx to communicate with PHP**

Open configuration file and add the following code in location directive

location ~ \.php$ {

include snippets/fastcgi-php.conf; # for security reason

fastcgi\_pass unix:/var/run/php/php7.2-fpm.sock; # for communication over sock not TCP

fastcgi\_intercept\_errors on;

}

save & test the configuration file then reload the configuration

now create a phpinfo.php file in /var/www/wisdompetmed.local and test the php settings

**Installing and Configuration Mariadb (018)**

install mariadb using following command

apt-get install mariadb-server mariadb-client

check the mariadb services

systemctl status mysqld.service

check the version of mariadb

mysql --version

Now make secure the installation using following command

mysql\_secure\_installation run this command to make the db secure

connect to mysql using below command

mysql -u root -p

enter the password

create a new database using commands

create database appointments;

create a new user and grant full permission on appointment databases

create user ‘admin’@’localhost’ identified by ‘[m@g1t786](mailto:m@g1t786)’;

grant all on appointment.\* to ‘admin’@’localhost’;

flush privileges;

few useful mysql commands

show databases;

use appointments;

show tables;

**Adding Dynamic Content to test Mysql connectivity and PHP (019)**

To Check all application services status in one commands

systemctl status nginx mysqld php7.2-fpm | grep -E “(Loaded|Active)”

now create a directory with name **appointments** inside /var/www/wisdompedmed directory

copy the index.php file to appointment directory

change permission using chomod +r (755) index.php

now restore the database from file

mysql -u admin [-pm@g1t786](mailto:-pm@g1t786) appointments< sql\_file

mysql -u user -ppassword database\_name < file\_name

**Secure Site with Nginx (020)**

we should consider follow to secure website

1. make sure operating system is up to date

2. Restrict access where possible to certain part of website

3. use password protect sensitive data / information

4. use SSL transmissions and identity your site

**Configure Allow and Deny Directories (021)**

we want to secure the appointments/index.php file

create a location for appointment in using following configuration

**location /appointments/ {**

**allow 192.168.20.0/24;** # enter IP address by which you are accessing the site.

**allow 10.0.0.0/8;**

**deny all;**

**}**

save configuration and test configuration and reload configurations

now try to access the index.php file in appointments , we will receive

403 Forbidden page

now to allow access to internal request to index.php file, add the IP range using

allow 192.168.20.0/24;

reload the page now it should load now Configure

Above configuration will deny access to appointment folder from any IP address except

192.168.20.0 address or subnet.

**Creating 403 page (022)**

To display a proper page in place of 403 default page.

edit configuration file and add deny location

location /deny {

deny all;

}

test using 192.168.0.3/deny and make sure you are getting default 403 page

add the below configuration to file to display 403.html page

location =deny {

deny all;

}

error\_page 403 /403.html;

location =403.html {

internal;

}

now copy the 404.html and modify for 403.html page in directory

**Configure Password Authentication for specific Location in config file (023)**

we will use htpasswd utility to create password file

this utility with apache

install it using command **apt-get install apache2-utils -y**

now create password file

htpasswd -c /etc/nginx/passwords admin # this command will create file with user name admin and prmot for password

to add new user, run the above command without -c flag, otherwise it will overwrite it

to change user password run the above command without c flag

to remove the user use the -D flag to delete the user

change the owner and permission on password file

chown www-data /etc/nginx/passwords

chmod 600 /etc/nginx/passwords

now edit the configuration file to use the password

add following command in appointments locations directives

auth\_basic “Authentication is required ...”;

auth\_basic\_user\_file /etc/nginx/passwords;

location ~ \.php$ {

include snippets/fastcgi-php.conf;

fastcgi\_pass unix:/var/run/php/php7.2-fpm.sock;

}

**Configuring HTTPS (024) &**

SSL Secure sock layer this is used to encrypt the traffic. Previously used this method

TLS Transport layer certification this is new encryption, prefer over SSL

**Create an SSL Certificate (025)**

Creating Self Sign Certificate

openssl command is used to generate the local certificate

apt install openssl

**openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/nginx.key -out /etc/ssl/certs/nginx.crt -bash**

req request

-x509 x509 is type of certificate

nodes not used DES encryption method

-days 365 valid for 1 year

-newkey to generate new key

rsa:2048 RSA is encryption method with 2048 bit key

-keyout path to store key

-out path to store certificate

-batch this is optional flag, not ask for information and press enter

Certificate file will be public, but the key need to private and should not be expose

Now we will add certificate to Nginx

**Install SSL certificate on NGINX (026)**

now edit the configuration to add the SSL certificate to site-enable

add the following to config file a top of configuration file

server {

listen 80 default\_server;

returning 301 https://$server\_addr$request\_uri;

}

this server directive will instruction request coming on port 80 and redirect to port 443 port with return directive. 301 code mean traffic on port 80 will permanently redirect to port 443.

now change the old server directive port from 80 to 443 and add ssl direct as well like below code

server {

listen 443 ssl default\_server;

ssl\_certificate /etc/ssl/certs/nginx.crt;

ssl\_certificate\_key /etc/ssl/private/nginx.key;

}

now save the configuration and test and reload and check with browser.

**Reverse Proxies and Load Balancing (027)**

both proxy and load balancing are almost same the only difference is that proxy is using with one additional server while load balance is more than one server, LB also provide session persistence as well

Proxy site between client and Server → request from client first go to proxy then proxy send the request to server.

**Configure Nginx as proxy server (028)**

Upstream module is used for proxy and load balancing server

create another configuration file with name upstream.conf in /etc/nginx/conf.d directive and add following code in it

upstream app\_server\_7001 {

server 127.0.0.1:7001;

}

server {

listen 80;

location /proxy {

# trailing slash is must

proxy\_pass http:// **app\_server\_7001**/;

}

}

Server will listen on port 80, when request will come, server will forward the request to proxy and give it to upstream.

now save the configuration and test and reload and check with browser.

**Configure Nginx as Load Balancer (029)**

Nginx use three methods for load balancing

1. Round Robin default method no preferences to any server, on server connected at a time

2. Least Connections server with fewer request is favored

3. IP Hash session persistence and connection is made based on client IP address

4. All this is weighted directives, server with higher weighted is favored

now open the upstream.conf configuration file created in early lesson and add the following code for load balancing

add upstream block and gave name round Robin

upstream app\_server\_7001 {

server 127.0.0.1:7001;

}

upstream roundrobin {

server 127.0.0.1:7001;

server 127.0.0.1:7002;

server 127.0.0.1:7003;

}

upstream leastconn {

least\_conn;

server 127.0.0.1:7001;

server 127.0.0.1:7002;

server 127.0.0.1:7003;

}

upstream iphash {

ip\_hash;

server 127.0.0.1:7001;

server 127.0.0.1:7002;

server 127.0.0.1:7003;

}

upstream weighted {

server 127.0.0.1:7001 weight=2;

server 127.0.0.1:7002;

server 127.0.0.1:7003;

}

server {

listen 80;

location /proxy {

proxy\_pass http:// **app\_server\_7001**/;

}

location /roundrobin/ {

proxy\_pass http://roundrobin/;

}

location /roundrobin/ {

proxy\_pass http://leastconn/;

}

location /roundrobin/ {

proxy\_pass http://iphash/;

}

location /roundrobin/ {

proxy\_pass http://weighted/;

}

}

save the file and test the configurations by check 192.168.0.3/roundrobin

now for other method follow the process below, copy the roundrobin upstream and paste three time

upstream leastconn {

least\_conn;

server 127.0.0.1:7001;

server 127.0.0.1:7002;

server 127.0.0.1:7003;

}

upstream iphash {

ip\_hash;

server 127.0.0.1:7001;

server 127.0.0.1:7002;

server 127.0.0.1:7003;

}

upstream weighted {

server 127.0.0.1:7001 weight=2;

server 127.0.0.1:7002;

server 127.0.0.1:7003;

}

and add location for each upstream, copy the roundrobin location paste three time

location / leastconn {

proxy\_pass http://leastconn/;

}

location / iphash {

proxy\_pass http://iphash/;

}

location / weighted {

proxy\_pass http://weighted/;

}

now save the file and test the configuration by reloading and in browser

192.168.0.3/leastconn

192.168.0.3/iphash

192.168.0.3/ weighted

**Example No. 1 Hosting Simple Website**

1. Install Nginx and test
2. Create a folder in /var/www/html
3. Copy the HTML, CSS, Images to newly created directory
4. Now create nginx virtual host
5. Create a file in /etc/nginx/sites-available/simplewebsite and enter following code

server {

listen 80;

server\_name 192.168.10.10;

index index.html index.php index.htm;

root /var/www/html/simplewebsite;

}

1. Create system link to file

ln -s /etc/ngix/sites-available/simplewebsite /etc/nginx/sites-enable/

1. Restart the nginx and test site from browser

**Example No. 2 Hosting Simple Multiple Website using single Nginx configuration file**

1. Install Nginx and test
2. Create a two folders in /var/www/html/ with name site1 and site2
3. Copy the HTML, CSS, Images to newly created directory repectively
4. Now create nginx virtual host
5. Create a file in /etc/nginx/sites-available/simplewebsite and enter following code

server {

listen 80;

server\_name 192.168.10.10;

index index.html index.php index.htm;

root /var/www/html/site1;

}

listen 8080;

server\_name 192.168.10.10;

index index.html index.php index.htm;

root /var/www/html/site2;

}

1. Create system link to file

ln -s /etc/ngix/sites-available/simplewebsite /etc/nginx/sites-enable/

1. Restart the nginx and test site from browser, now we have two sites, one is running on port 8 and another on port 8080

**Example No. 3 Hosting Simple Website with location context and modifiers**

1. Install Nginx and test
2. Create a folder in /var/www/html with name site3
3. Copy the HTML, CSS, Images to newly created directory
4. Now create nginx virtual host
5. Create a file in /etc/nginx/sites-available/simplewebsite and enter following code

server {

listen 80;

server\_name 192.168.10.10;

index index.html index.php index.htm;

root /var/www/html/site3;

# / mean anything match, /contact will match anything start with contact, and also not case sensitive

location /contact {

return 200 ‘This is “/contact” location from nginx’;

}

# = means exact match and is case sensitive

location = /contact {

return 200 ‘This is “/contact” location from nginx’;

}

# ~ means regular expression and is case sensitive, if with staric \* like ~\* make case insensitive and ^~ with give preference to location

location ~ /contact[0-9] {

return 200 ‘This is “/contact” location from nginx’;

}

}

1. Create system link to file

ln -s /etc/ngix/sites-available/simplewebsite /etc/nginx/sites-enable/

1. Restart the nginx and test site from browser

**Example No. 4 Hosting Simple Website with PHP support and troubleshooting**

1. Install Nginx & PHP-FPM and test
2. To verify PHP is running issue below command

**systemctl list-units | grep php**

**systemctl status php7.4-fpm**

1. Create a folder in /var/www/html
2. Copy the HTML, CSS, Images to newly created directory
3. Now create nginx virtual host
4. Create a file in /etc/nginx/sites-available/simplewebsite and enter following code

server {

listen 80;

server\_name 192.168.10.10;

index index.php index.html index.htm;

root /var/www/html/simplewebsite;

# below location will process all request and match with /uri, if found will work otherwise return 404 page

location / {

try\_fiiles $uri $uri/ =404;

}

# below location will match all file having .php extension

location ~ /.php$ {

include fastcgi.conf; # this file is by default included

fastcgi\_pass unix: /run/php/php7.4-fpm.sock; # to search for find / -name \*fpm.sock

}

}

1. Now create a php file in root directory
2. Create system link to file

ln -s /etc/ngix/sites-available/simplewebsite /etc/nginx/sites-enable/

1. Restart the nginx and test site from browser

Troubleshooting

1. If we getting 502 bad gateway error

Read the last line of error file with below command

tail -n 1 /var/log/nginx/error.log

1. Check the user, for nginx services using below command

ps aux | grep nginx

1. Check the user, for php services using below command

ps aux | grep nginx

make sure both process are running using www-data user

Install & Configure Let’s Encrypt SSL Certificate

Let’s encrypt is certificate authority that provide free X.509 certificates for TLS encryption

Certificate is valid for 90 days / 3 months

1. Login to server
2. Install certbot using below command

add-apt-repository ppa:certbot/certbot

1. Install certbot nginx package using below command

apt-get install python3-certbot-nginx

choose Y, N & N option

1. Now obtain SSL Certificate using below command

certbot --nginx -d 192.168.10.10

enter email address and press A , press Y, Choose option No. 2

1. How to renew the certificate for 1 year, automatically

certbot renew --dry

echo | openssl s\_client -connect your\_domain\_name:443 2>/dev/null | openssl x509 -noout -dates

1. Use the below link to generate the SSL Configuration

https://ssl-config.mozilla.org/#server=nginx&version=1.18.0&config=modern&openssl=1.1.1d&guideline=5.6